

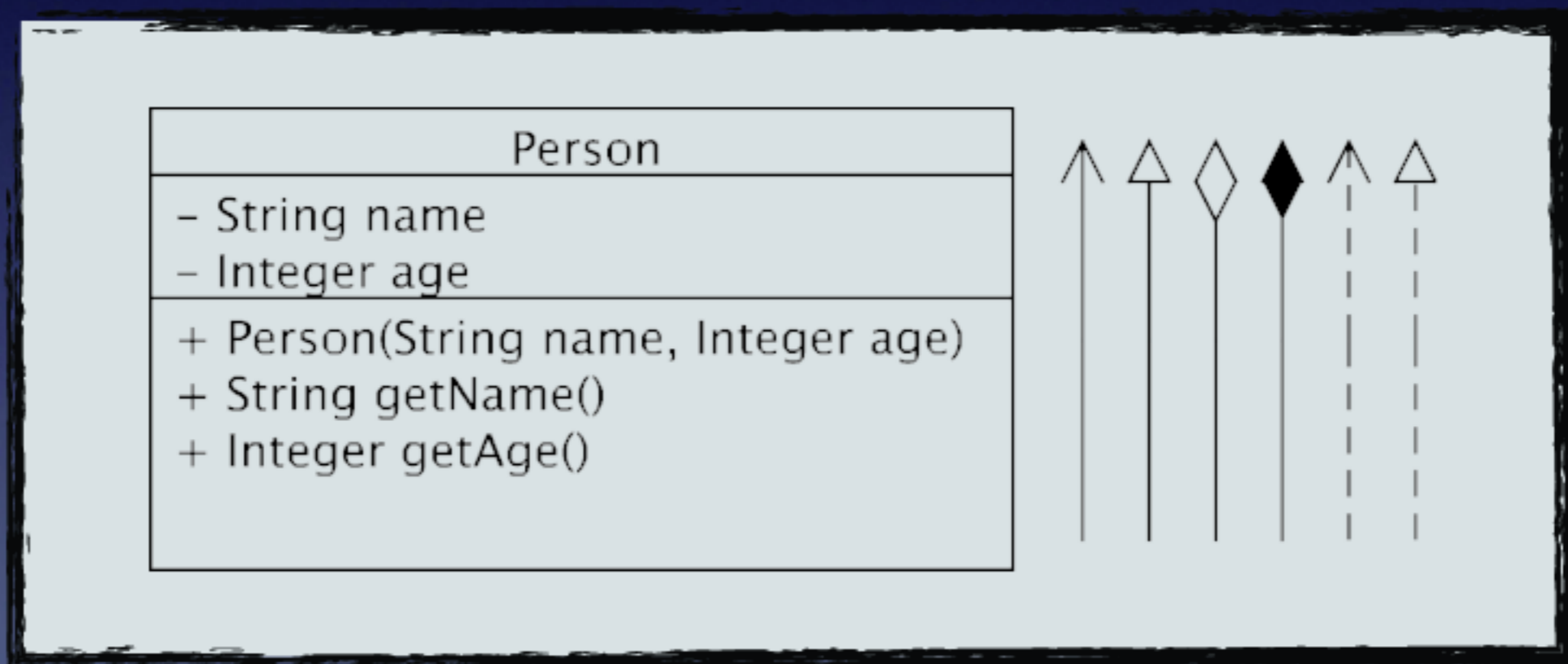


MVC

Skönhet sitter på utsidan och är löskopplad från intelligensen. Boken döms alltid efter omslaget.

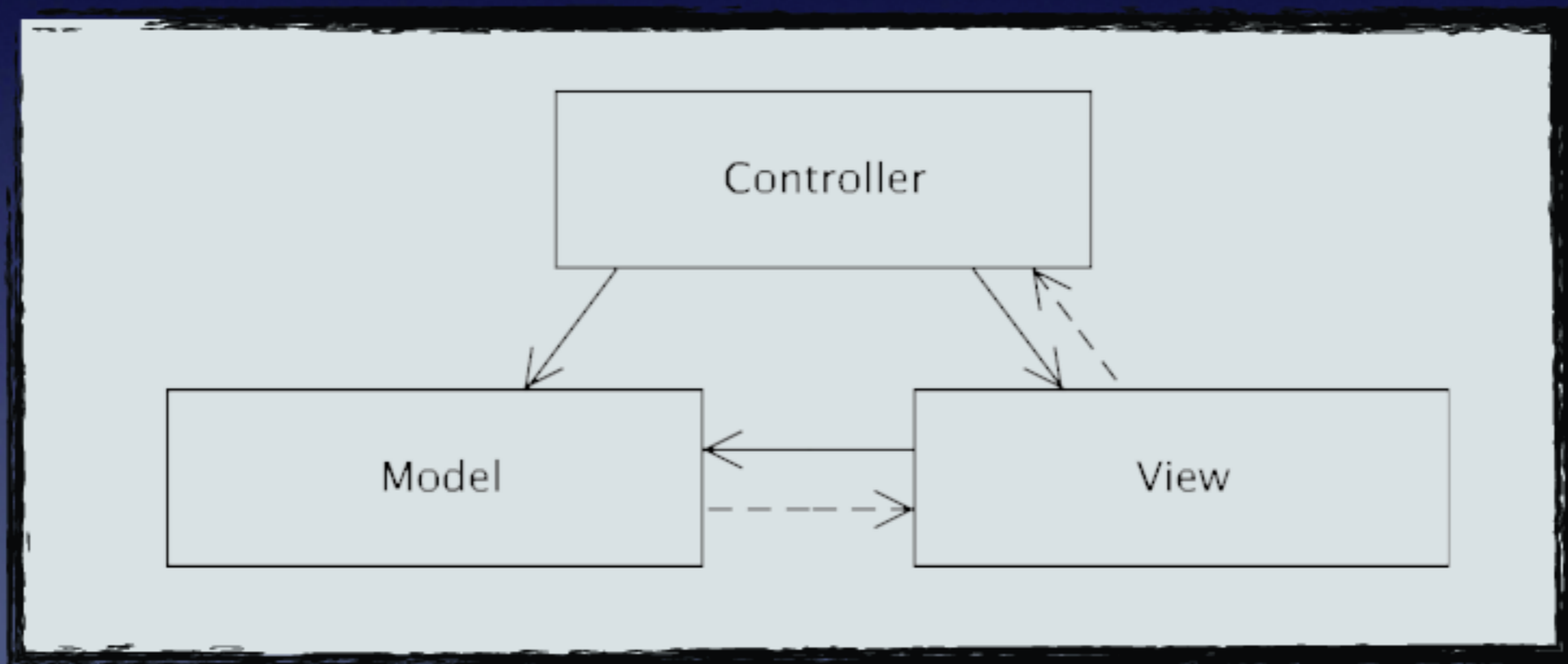
UML

- Grafisk representation av mjukvarudesign
- Används vid dokumentation och planering



MVC

- Separerar logik från utseende och data
- Användbart när man bygger grafiska applikationer, exempelvis hemsidor



<http://testa.roboto.se/>

Titel:

--	--

Spara

Editeringslänk: <http://testa.roboro.se/edit/45312>

Testlänk: <http://testa.roboro.se/74897>

Titel:

Engelska

Hund
Katt
Strumpor

Dog
Cat
Socks

Spara

Editeringslänk: <http://testa.roboro.se/edit/45312>

Testlänk: <http://testa.roboro.se/74897>

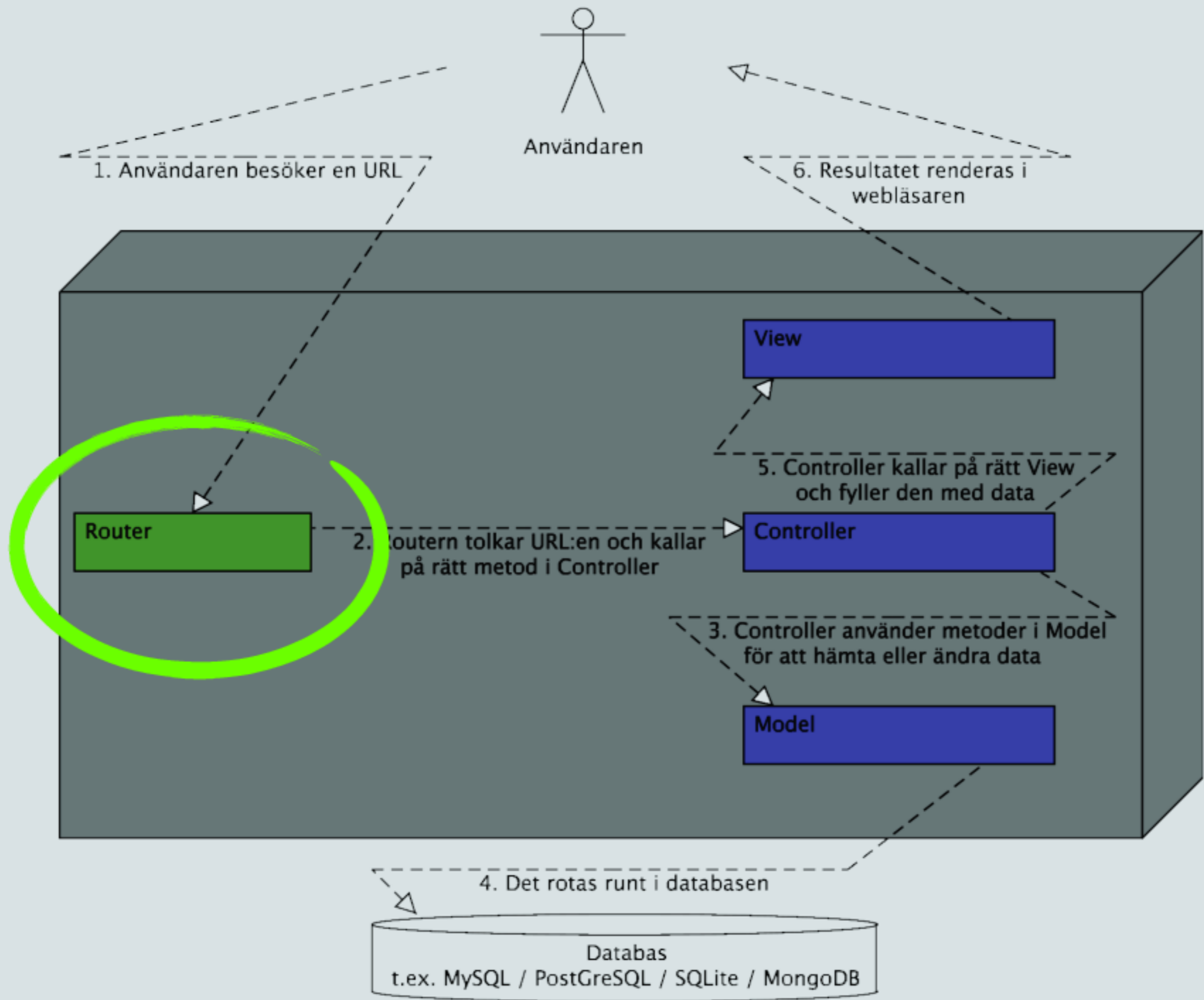
Engelska

Hund

Katt

Strumpor

Rätta!



Routeren

- Tolkar URL:er
- Skickar användaren till rätt kontroller
- Består i detta fall av två filer:
 - .htaccess
 - index.php

.htaccess i Apache

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?$1 [L,QSA]
```

<http://testa.roboro.se/index.php?edit/523>

<http://testa.roboro.se/edit/523>

Apache skickar "edit/523" till index.php

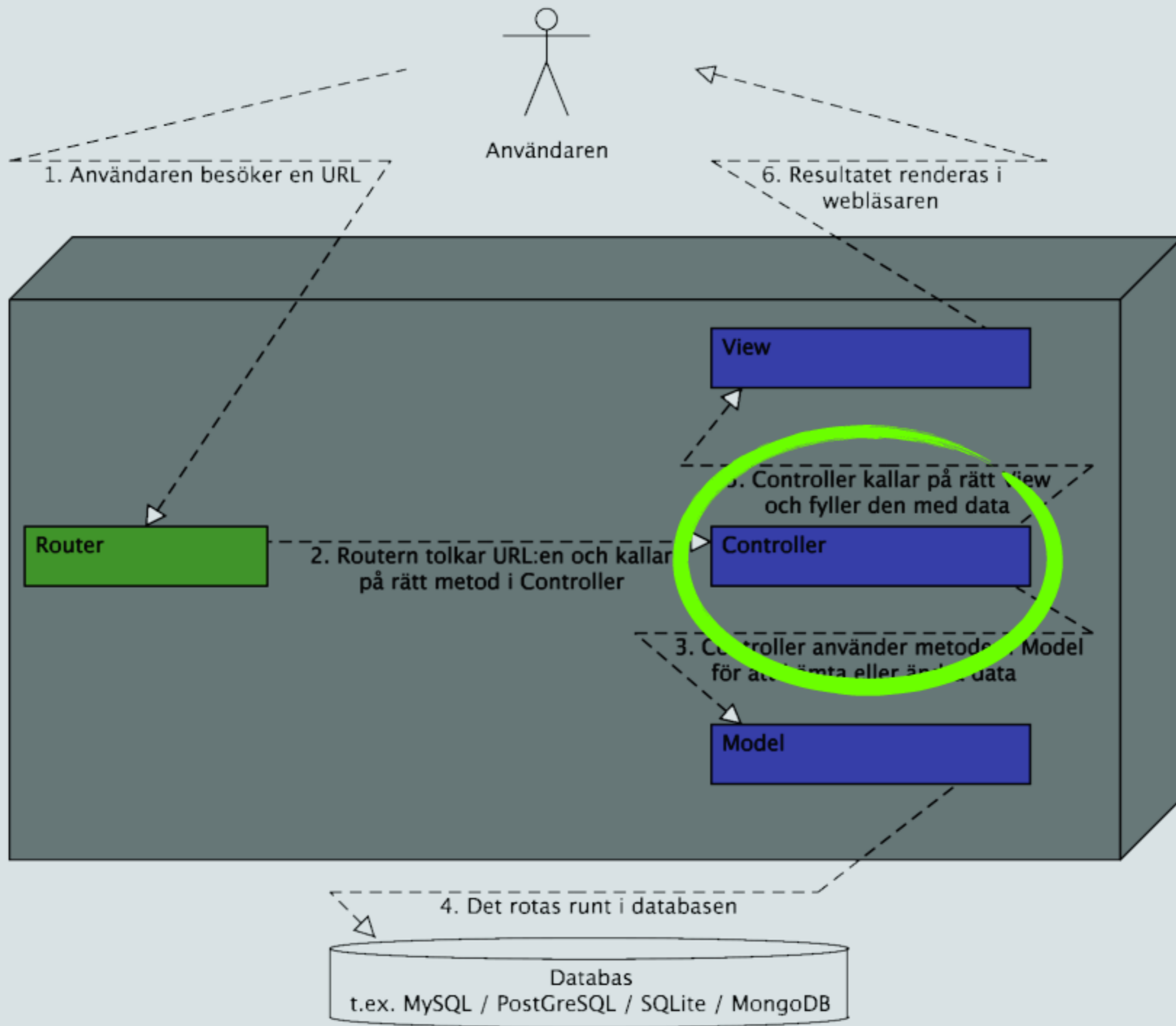
index.php

```
chdir( '../application' );  
  
include 'model/database.php';  
  
foreach (glob( 'controller/*.php' ) as $controller)  
    include $controller;  
  
$query = $_SERVER[ 'QUERY_STRING' ];
```

url-matching i index.php

```
$routes = array
(
    '/^$/'           => function($m) { control_new(); },
    '/^edit\//(\d+)$/' => function($m) { control_edit($m[1]); },
    '/^(\d+)$/'      => function($m) { control_test($m[1]); },
    '/.*$/'          => function($m) { control_error('Konstig sida. '); }
);

foreach ($routes as $regex => $function)
{
    if (preg_match($regex, $query, $matches))
    {
        $function($matches);
        break;
    }
}
```



Kontrollern

- Funktioner som kallas på av routern
- Anropar funktioner i modellen
- Förbereder data åt vyer
- Anropar rätt vy

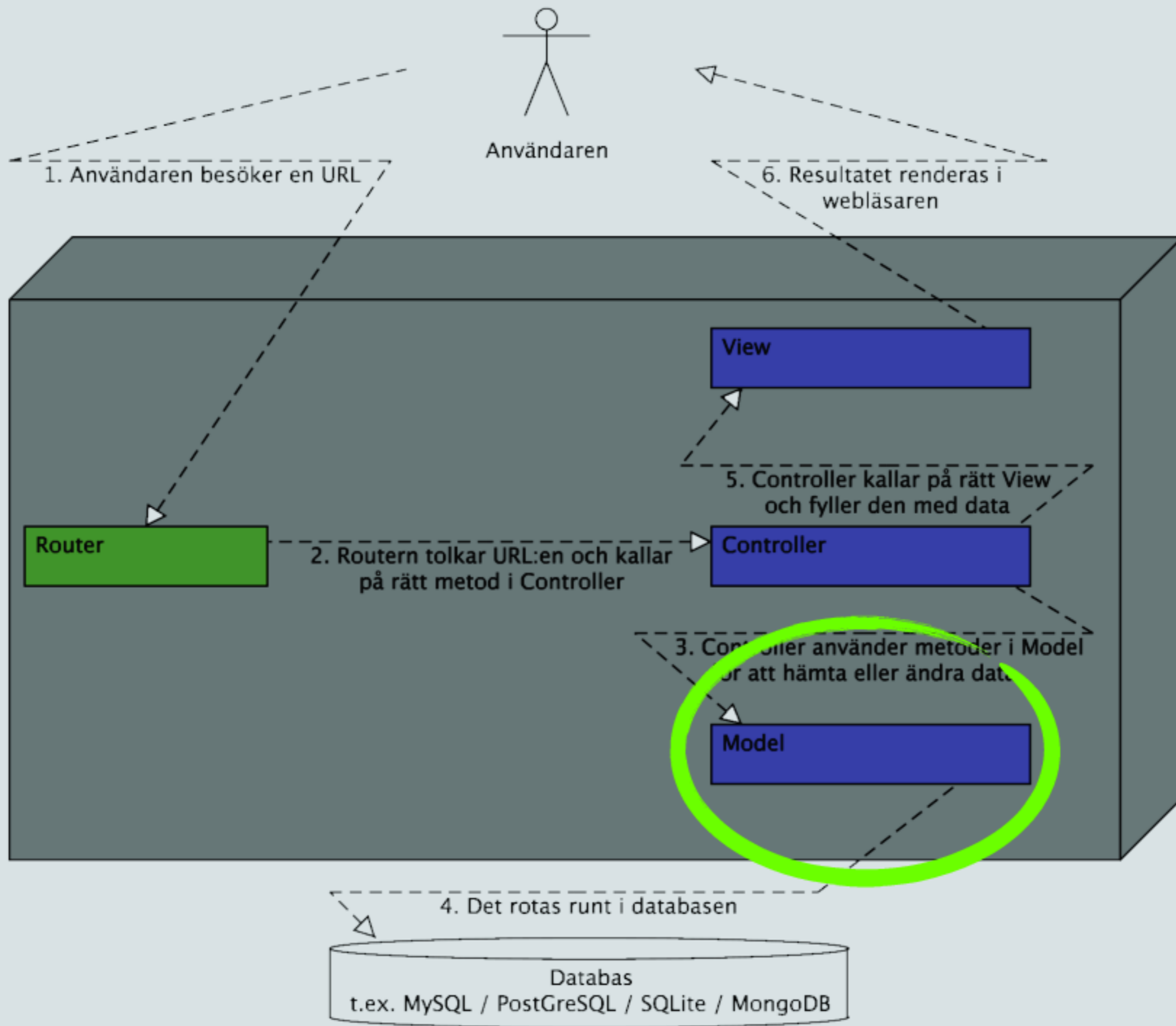
Exempelkontroller

```
function control_new()  
{  
    $db = Database::get_instance();  
    $editId = $db->create_test();  
    header('Location: /edit/'. $editId);  
}
```

Exempelkontroller

```
function control_error($message)
{
    $title          = "Error i modermodemet";
    $stylesheet     = "error";

    include 'view/header.php.html';
    include 'view/error.php.html';
    include 'view/footer.php.html';
}
```

Användaren

1. Användaren besöker en URL

6. Resultatet renderas i webbläsaren

Router

2. Routern tolkar URL:en och kallar på rätt metod i Controller

View

Controller

5. Controller kallar på rätt View och fyller den med data

3. Controller använder metoder i Model för att hämta eller ändra data

Model

4. Det rotas runt i databasen

Databas
t.ex. MySQL / PostgreSQL / SQLite / MongoDB

Modellen

- Innehåller applikationens data
- Tillhandahåller ett gränssnitt mot data
- Hanterar ofta någon form av databas
- Vi skrev en databasklass så att koden blev tjugisig

```

Database
(
    private $db;
    private static $instance = null;

    public static function get_instance()
    {
        if (self::$instance == null)
            self::$instance = new Database();

        return self::$instance;
    }

    private function Database()
    {
        $db_file = 'database/database.sqlite';

        if (! file_exists($db_file))
        {
            $this->db = new PDO('sqlite:'. $db_file);

            $this->db->query('CREATE TABLE Test
                (id INTEGER PRIMARY KEY,
                 editId INTEGER,
                 testId INTEGER,
                 title TEXT,
                 questions TEXT,
                 answers TEXT)');
        }
        else
        {
            $this->db = new PDO('sqlite:'. $db_file);
        }
    }

    function update_test($editId, $title, $questions, $answers)
    {
        $statement =
            $this->db->prepare('UPDATE Test
                set title = ?,
                   questions = ?,
                   answers = ?
                WHERE editid = ?');

        $statement->execute(array($title, $questions, $answers, $editid));
    }

    function get_test_by_editid($editid)
    {
        $statement =
            $this->db->prepare('SELECT title, questions, answers, testid, editid
                FROM Test
                WHERE editid = ?');

        $statement->execute(array($editid));

        return $statement->fetch();
    }
}

```

```

function get_test_by_testid($testid)
{
    $statement =
        $this->db->prepare('SELECT title, questions, answers
            FROM Test
            WHERE testid = ?');

    $statement->execute(array($testid));

    return $statement->fetch();
}

function create_test()
{
    do
    {
        $editid = rand(10000, 99999);
    } while ($this->get_test_by_editid($editid));

    do
    {
        $testid = rand(10000, 99999);
    } while ($this->get_test_by_testid($testid));

    $statement =
        $this->db->prepare('INSERT INTO Test (editid,
            testid,
            title,
            questions,
            answers)
            VALUES (?, ?, "", "", "")');

    $statement->execute(array($editid, $testid));

    return $editid;
}
}

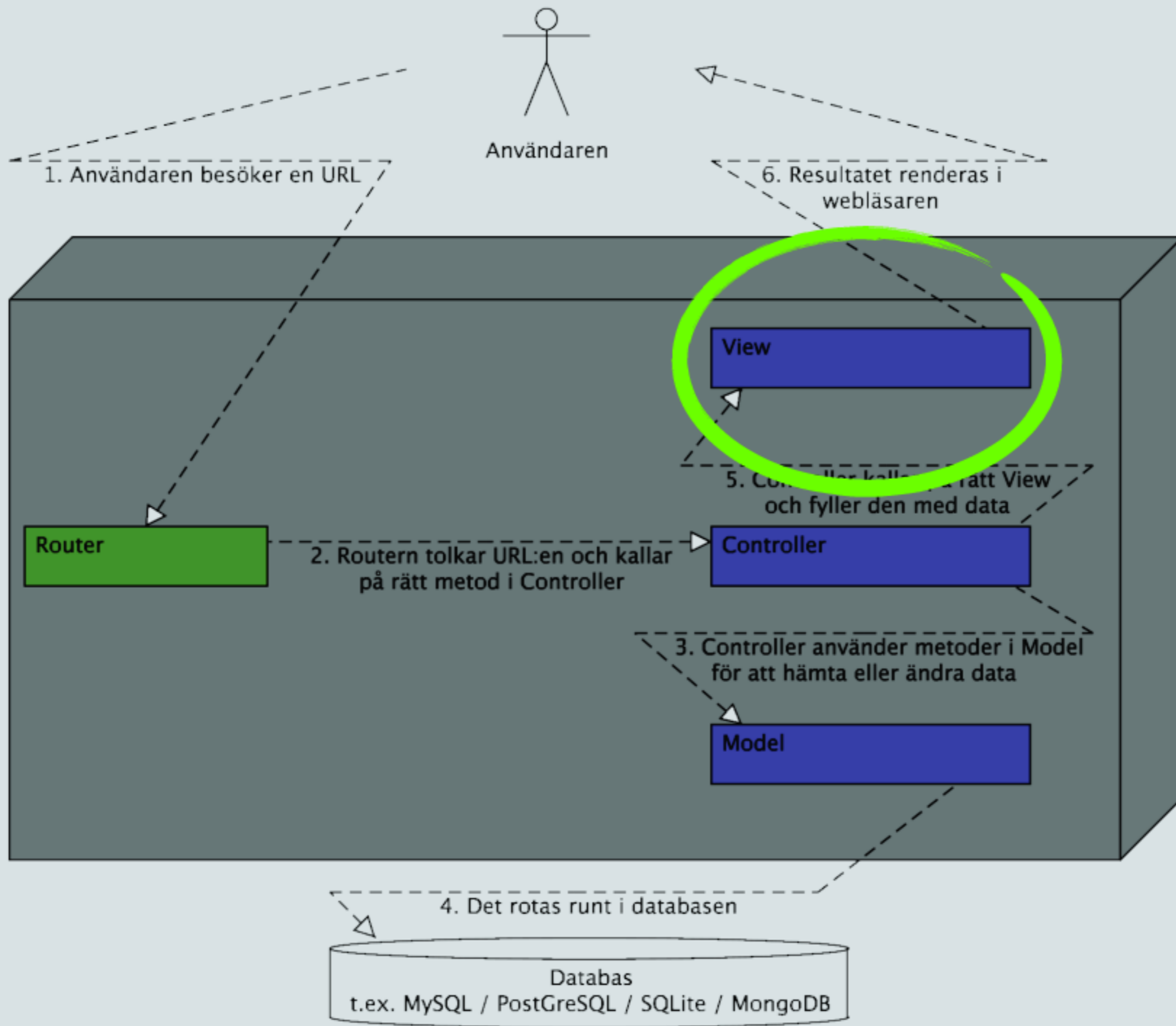
```

Exempelfunktion i modellerna

```
function get_test_by_testId($testId)
{
    $statement =
        $this->db->prepare('SELECT title,
                            questions,
                            answers
                            FROM Test
                            WHERE testId = ?');

    $statement->execute(array($testId));

    return $statement->fetch();
}
```



Vyn

- Vyer presenterar färdigställd data
- Sköter kommunikation med användaren

test.html.php

```
<h1><?= $title ?></h1>
<form action="" method="POST">
  <table>
<? foreach ($test_questions as $test_question) { ?>
  <tr>
    <td>
      <?= $test_question['question'] ?>
    </td>
    <td>
      <input type="text"
        name="answers[]"
        value="<?= $test_question['user_answer'] ?>" />
      <?= $test_question['is_correct'] ? "" : "FEL!" ?>
    </td>
  </tr>
<? } ?>
  </table>
  <input type="submit" name="submit" value="Rätta!" />
</form>
```