# Резюме на дипломен проект

# Представяне

Уважаема г-жо Директор, Уважаема комисия, уважаеми гости.

Аз съм Мирослава Тодева

Темата на моят проект е „Удостоверяване на самоличността и оторизация в уеб приложения.“

# Въведение в уеб сигурността

В днешно време уеб сигурността е едно от най-важните неща, когато говорим за защита на личните данни и гарантиране на поверителността в интернет. Всеки ден ние използваме различни уеб приложения и сайтове, и за да сме сигурни, че само упълномощените хора имат достъп до тях, се използват различни методи за удостоверяване и оторизация.

# Ключови процеси – удостоверяване и оторизация

Удостоверяването е процесът, чрез който системата проверява кой си ти. Обикновено това става чрез въвеждане на потребителско име и парола, но има и по-сложни начини като двуфакторна идентификация, където трябва да въведеш код, изпратен по SMS, или биометрични данни като пръстов отпечатък. Важното е потребителят да докаже, че именно той е човека, който иска достъп.

# Значение на сигурността в уеб приложенията

След като удостоверим потребителя, идва ред на оторизацията. Тя определя какви ресурси или действия му позволяваме да извърши. Например, някой може да има право само да гледа данни, а друг – да ги редактира или изтрива. Влезлият в системата може да има различни роли: гост, обикновен потребител или администратор. Всяка роля има различни права, като това се нарича ролеви базиран достъп.

# Основни подходи за удостоверяване

При реализацията на уеб приложения тези процеси обикновено се грижат за тях с помощта на специални протоколи и технологии. Например, използваме бисквитки или JSON Web Token, за да запомним кой е влязъл и какви права има. За по-сигурна комуникация често се ползват сертификати и различни стандарти като OAuth или SSO, които улесняват потребителите и същевременно изграждат надеждна защита.

# Механизми и протоколи за удостоверяване

Въпреки това, интернет е пълен и с опасности. Лошите хора използват слабости като слаби пароли, SQL инжекции или XSS атаки, за да проникнат в системите. Затова е много важно всичко да се защитава правилно: да се използват силни пароли, криптиране и двуфакторна автентикация. Тази сигурност е важна не само за личната защита, но и за защитата на цялата система.

# Типични уязвимости и добри практики

Интернет системите често са уязвими към слабости като слаби пароли, SQL инжекции, XSS атаки и други. За да се предпазим от това , е важно да използваме силни пароли, криптиране и двуфакторна автентикация, което осигурява по-висока сигурност както за личната информация, така и за цялата система.

# Процес на оторизация

Ролева (RBAC) – роли и права, например администратор, потребител

Атрибутна (ABAC) – основана на атрибути като час, местоположение

Примери за нива на достъп: гост, регистриран, администратор.

Оторизацията определя правата на потребителя след удостоверяване. Системи използват ролева (RBAC) или атрибутна (ABAC) модели. Примери за нива на достъп са гост, регистриран или администратор, всеки със собствен набор от права.

# Архитектурни модели и стандарти

Използват се механизми като сесии, бисквитки и JSON Web Tokens (JWT) за управление на удостоверяването и достъпа. Възможно е използване на различни библиотеки и рамки като Django системата за удостоверяване и авторизация.

# Средства за удостоверяване и оторизация в Django

Django разполага с вградена система django.contrib.auth, която предлага модели за потребители, групи и права. Има готови форми за логин и регистрация, както и методи за защита чрез декоратори като @login\_required и @permission\_required.

# Практическа реализация

В рамките на проекта беше създадено уеб приложение с използване на Django framework, което демонстрира процеса на удостоверяване и оторизация на потребителите. Реализирани бяха функции за регистрация, вход, управление на профил и контрол на достъпа до различни части на системата чрез роли и права. Използваха се стандартни компоненти като форми за вход, сесии и бисквитки за съхраняване на състоянието. Освен това, беше внедрена система за защита срещу типични уязвимости като SQL инжекции и XSS атаки, чрез проверка и валидиране на входните данни, както и използване на криптиране и хеширане на пароли.

# Обща архитектура на приложението

Приложението е базирано на тропична клиент-сървър архитектура, реализирана с Django като сървъчен сайт и HTML/CSS/JavaScript като клиентски интерфейс. Backend системата включва модели за потребители, роли и права, както и логика за удостоверяване и контрол на достъпа. Данните се съхраняват в релационна база данни (например PostgreSQL), която поддържа връзки между таблиците и гарантира цялостност и сигурност на информацията. Клиентската част комуникира с сървъра чрез HTTP заявки, използвайки REST API или стандартни Django изгледи, като при успешна автентикация се използват бисквитки или JSON Web Tokens за удостоверяване на потребителя. Структурата е модулна и разширяема, което позволява лесно добавяне на нови функции и подобряване на сигурността.

# Заключение

Обобщавайки всичко, можем да кажем, че сигурността на уеб приложенията е нещо изключително важно и трябва да се поддържа постоянно. Трябва да използваме всички съвременни методи и технологии, както и да обучаваме потребителите как да защитават себе си. Това е не само бъдещето на интернет, но и отговорност на всеки, който работи с такива системи.