

# **Studytask Project Plan**

## **1) Elevator Pitch**

StudyTasks is a simple website for students at FH Technikum Wien. It helps students organize study tasks and deadlines and work together in small project groups. The focus is on simplicity, overview, and teamwork.

## **2) Target Group**

- Students at FH Technikum Wien
- Especially first-year students who have many projects and deadlines
  
- Main problems:
  - Hard to keep track of deadlines and tasks
  - Unclear who does what in group work
  - No simple and central website for task management

## **3) Main Goals**

1. Give a clear overview of all upcoming tasks and deadlines
2. Make teamwork and group coordination easier
3. Help students plan their time and meet deadlines

## **4) Core Features (simple and realistic)**

Main Features:

- Task Management
  - create, edit, delete, and mark tasks as done
- Deadline Overview
  - simple calendar that shows all upcoming tasks
- Group Management & Overview
  - create groups, manage members, assign tasks to team members and show status (“To Do”, “In Progress”, “Done”)
- Reminder System – automatic notifications before deadlines
- Progress Dashboard – shows completed and open tasks

Optional Feature: - Tagging System – sort and filter tasks by subject or project

## 5) Functional Requirements (Gherkin, all features)

- User registration & login
- Task management (CRUD)
- Deadlines & calendar view
- Group management (1 group per user, join via code, owner controls)
- Task status & claiming (one responsible person per task)
- Reminder system (upcoming deadline alerts)
- Progress dashboard
- Tags (optional)

**Roles:** Guest, Student (Member), Owner

### **Global Rules:**

Rule: Single group constraint

A user may belong to only one group at a time.

Rule: Single task owner

A task may have at most one responsible person.

Rule: Visibility

Group and task data is visible only to group members.

---

### **FR1 – Registration & Login (Must)**

Feature: Registration & Login

Scenario: Register

Given I am a guest

When I register with email and password

Then a user account is created

And I am logged in

Scenario: Login

Given I have an existing account

When I enter valid login credentials

Then I am logged in

---

### **FR2 – Create, Edit, Delete Tasks (Must)**

Feature: Task CRUD

Scenario: Create a task

Given I am a group member

When I create a task with title, deadline, and status "open"

Then the task is saved

And it appears in the task list

Scenario: Edit a task

Given a task exists

When I update the title or deadline

Then the changes are saved

Scenario: Delete a task

Given a task exists

When I delete it

Then it is removed from the task list

---

### **FR3 – Task Attributes (Must)**

Feature: Task fields

Every task must have:

- \* Title

- \* Deadline

- \* Status (open, in progress, done)

---

### **FR4 – Weekly Calendar View (Must)**

Feature: Weekly view

Scenario: View weekly tasks

Given tasks with deadlines exist

When I open the weekly calendar view

Then I see all tasks for that week with deadlines and status

---

### **FR5 – Assign / Claim Tasks (Should/Must per updated scope)**

Feature: Task claiming

Scenario: Claim an unassigned task

Given a task has no responsible user

When I claim it

Then I become the responsible person

Scenario: Prevent double claim

Given a task already has a responsible user

When I attempt to claim it

Then I receive an error message

Scenario: Unclaim a task

Given I am the responsible person

When I unclaim the task

Then the task has no responsible person

---

### **FR6 – Deadline Reminders (Should)**

Feature: Deadline reminders

Scenario: Receive reminder

Given a task has a deadline

When the deadline approaches

Then I receive a notification (UI/email/popup)

---

### **FR7 – Progress Dashboard (Should)**

Feature: Progress dashboard

Scenario: View progress

Given tasks exist

When I open the dashboard

Then I see completed vs open tasks

---

### **FR8 – Tag System (Could)**

Feature: Tags

Scenario: Tag a task

Given a task exists

When I add a tag such as "#Math"

Then the task displays this tag

---

### **FR9 – Create & Join Group (Must)**

Feature: Create group

Scenario: Create

Given I am logged in and not in a group

When I create a group

Then I become the owner

And a join code is generated

Feature: Join group

Scenario: Join via code

Given I am not in a group

When I enter a valid join code

Then I join the group

---

### **FR10 – Group Overview & Member Management (Must)**

Feature: View members

Scenario: Member list

Given I am a group member

When I open the member page

Then I see all members with names and roles

Feature: Kick members (Owner)

Scenario: Remove a member

Given I am the owner

When I choose "Remove member"

Then the member is removed from the group

---

### **FR11 – Leave Group & Transfer Ownership (Must)**

Feature: Leave group

Scenario: Member leaves

When I click "Leave group"

Then I am removed

Scenario: Owner leaves

Given I am the owner

When I choose to leave

Then I must transfer ownership first

---

### Non-Functional Requirements

NFR: Load time < 5s in modern desktop browsers

NFR: Persistent database storage

NFR: Simple and clear UI

---

### Acceptance Checklist

- Login/registration
- One group per user
- Join via code
- View/kick/transfer members
- Task CRUD
- Weekly calendar
- Task claiming
- Deadline reminders
- Dashboard
- Optional tags

## 6) Non-Functional Requirements

Feature: Website performance

Scenario: Fast loading time

Given I am using the StudyTasks web application

And I am accessing it on a modern desktop browser

When I open the dashboard or any core page

Then the page must fully load in under 5 seconds

Feature: Browser compatibility

Scenario: App works in supported desktop browsers

Given I am using Chrome, Firefox, Edge, or Safari

When I access and use the StudyTasks system

Then all features must work correctly

And no mobile app is required

Feature: Data security

Scenario: Data is stored safely

Given user data exists (tasks, deadlines, group information, claims)

When the system stores the information

Then the data must be saved securely in the database

And data must not be lost or exposed

Feature: Usability

Scenario: Simple and intuitive interface

Given I am a student using the platform

When I navigate through the core functionalities

| Tasks Page |

| Calendar View |

| Group Page |

Then the interface must be simple, clear, and easy to understand

And I must be able to complete core actions without training

Feature: Maintainability and version control

Scenario: Work stored and versioned in DevOps

Given project files and documentation exist

When developers work on the system

Then all project materials must be stored in Azure DevOps

And changes must be version-tracked