# Modules

app

Main application entry point for the Cost Manager API. Sets up Express, connects to MongoDB, mounts routers, and starts the server.

db

Database connection setup using Mongoose.

controllers/aboutController

Controller for the GET /api/about endpoint. Returns a JSON array of team members' first and last names.

controllers/costController

Controller functions for handling cost-related API endpoints.

controllers/userController

Controller for the GET /api/users/:id endpoint. Retrieves a single user's details and computes their total cost.

models/Cost

Mongoose model for tracking individual cost entries.

models/User

Mongoose model for the User collection.

routes/about

Express router for the `/api/about` endpoint.

routes/costs

Express router for cost-related API endpoints.

routes/users

Express router for user-related API endpoints.

# app

Main application entry point for the Cost Manager API.Sets up Express, connects to MongoDB, mounts routers, and starts the server.

# db

Database connection setup using Mongoose.

### db~connectDB() ➡ `Promise.<void>`

Connects the application to MongoDB Atlas using Mongoose.Reads connection URI from process.env.MONGO_URI.On failure, logs the error and exits process.

**Kind**: inner method of `db`
**Throws**:

- `Error` If unable to connect to MongoDB.

# controllers/aboutController

Controller for the GET /api/about endpoint. Returns a JSON array of team members' first and last names.

### controllers/aboutController~aboutController(req, res) ➡ `void`

GET /api/aboutReturns a JSON array of team members' first and last names.Example response (status 200):[ { first_name: "Israel", last_name: "Peled" }]

**Kind**: inner method of `controllers/aboutController`

| Param | Type | Description |
| --- | --- | --- |
| req | `express.Request` | Express request object. |
| res | `express.Response` | Express response object; sends JSON array. |

# controllers/costController

Controller functions for handling cost-related API endpoints.

- controllers/costController
  - ~addCost(req, res) ➡ `Promise.<void>`
  - ~getMonthlyReport(req, res) ➡ `Promise.<void>`

### controllers/costController~addCost(req, res) ➡ `Promise.<void>`

POST /api/addAdds a new cost item.Body (`req.body`) must include: - description {string} - category {string} (one of "food","health","housing","sport","education") - userid {number} - sum {number} - date {string|Date} (optional)Responses: • 200: Returns saved cost JSON. • 400: Missing required fields. • 500: Database error.

**Kind**: inner method of `controllers/costController`

| Param | Type |
| --- | --- |
| req | `express.Request` |
| res | `express.Response` |

### controllers/costController~getMonthlyReport(req, res) ➡ `Promise.<void>`

GET /api/reportReturns monthly cost report for a user.Query (`req.query`) must include: - id {string|number} - year {string|number} - month {string|number}Success (200) returns JSON:{ userid: Number, year: Number, month: Number, costs: [ { food: [ { sum:Number, description:String, day:Number } ] }, { health: [ { sum:Number, description:String, day:Number } ] }, { housing: [] }, { sport: [ { sum:Number, description:String, day:Number } ] }, { education:[ { sum:Number, description:String, day:Number } ] } ]}Responses: • 200: Report JSON. • 400: Missing id/year/month. • 500: Database error.

**Kind**: inner method of `controllers/costController`

| Param | Type |
| --- | --- |
| req | `express.Request` |
| res | `express.Response` |

# controllers/userController

Controller for the GET /api/users/:id endpoint. Retrieves a single user's details and computes their total cost.

### controllers/userController~getUser(req, res) ➡ `Promise.<void>`

Retrieves details of a single user by ID and calculates the total cost.

**Kind**: inner method of `controllers/userController`
**Status**: 200 - OK with JSON of user's first_name, last_name, id, and total cost.
**Status**: 404 - Not Found if user with given ID does not exist.
**Status**: 500 - Server Error if aggregation or database query fails.

| Param | Type | Description |
| --- | --- | --- |
| req | `express.Request` | Express request object; expects req.params.id to be the user ID (string or number). |
| res | `express.Response` | Express response object; returns JSON object: { first_name: string, last_name: string, id: number, total: number } |

# models/Cost

Mongoose model for tracking individual cost entries.

- models/Cost
  - ~costSchema : `mongoose.Schema.<Cost>`
  - ~Cost : `Object`

### models/Cost~costSchema : `mongoose.Schema.<Cost>`

Mongoose schema for Cost collection.

**Kind**: inner constant of `models/Cost`

### models/Cost~Cost : `Object`

**Kind**: inner typedef of `models/Cost`
**Properties**

| Name | Type | Description |
| --- | --- | --- |
| description | `string` | Description of the cost item (required). |
| category | `string` | Category of the cost; one of "food", "health", "housing", "sport", "education" (required). |
| userid | `number` | ID of the user who incurred the cost (required). |
| sum | `number` | Amount of the cost (required, min: 0). |
| date | `Date` | Date when the cost was created; defaults to now. |

# models/User

Mongoose model for the User collection.


- models/User
  - ~UserSchema : `mongoose.Schema.<User>`
  - ~User : `Object`


### models/User~UserSchema : `mongoose.Schema.<User>`

Mongoose schema for User collection.

**Kind**: inner constant of `models/User`

### models/User~User : `Object`

**Kind**: inner typedef of `models/User`
**Properties**

| Name | Type | Description |
| --- | --- | --- |
| id | `number` | Unique user ID (required). |
| first_name | `string` | First name (required). |
| last_name | `string` | Last name (required). |
| birthday | `Date` | Birth date (required). |
| marital_status | `string` | One of “single”, “married”, “divorced”, “widowed” (required). |

# routes/about

Express router for the `/api/about` endpoint.

### routes/about~router : `express.Router`

Express router instance.

**Kind**: inner constant of `routes/about`

# routes/costs

Express router for cost-related API endpoints.

### routes/costs~router : `express.Router`

Express Router for `/api/add` and `/api/report`.

**Kind**: inner constant of `routes/costs`

# routes/users

Express router for user-related API endpoints.

### routes/users~router : `express.Router`

Express Router for `/api/users/:id`.

**Kind**: inner constant of `routes/users`